CHAPTER 7

SPEAKER IDENTIFICATION

ALGORITHMS
Chapter 7

SPEAKER IDENTIFICATION ALGORITHMS

7.1 INTRODUCTION

There are many Speaker Identification methods based on different models for speech. The identification performance mainly depends on the features selected and the way in which they are used. Most of the speaker recognition methods employ a pattern recognition approach, in which the template formulated from different speaker dependent features being the pattern descriptor. A decision or pattern classification is based on the different distance measures.

The three stages involved in a speaker recognition experiment are:

1. Selection of appropriate speaker dependent features and their extraction from the original speech.
2. Template creation from the features extracted
3. Decision making after finding distance measures between the templates.

The recognition methods differ either in all these stages or in any one of the steps.
In this method the features are selected on the basis of their contribution to the performance of the system. A few features are selected based on their established speaker dependency. Then each one of these are used in the experiment and performance is evaluated. The features which contribute less to the performance are knocked out. The measurement of each feature is also different in this method. Instead of taking the pitch period, which is a known speaker dependent feature, four intermediate measurements are taken in this method. Similarly some new measurements are also introduced.

Two methods are tried for speaker identification. One is, the Fixed Text Phoneme Model (FTPM) and the other is Similarity Measure Method. In the first approach an entropy measure of the phonemes is made use of. In the second approach, the feature measures are formed into a template and the similarity between templates are determined.

7.2 FIXED TEXT PHONEME MODEL

According to the speech model discussed in the previous chapter, there is an error pattern due to approximations and feedback systems. This error pattern must be unique for every speaker and must be determinable from measurable quantities. Since a fixed text approach is used in this system, the probability of
occurrence of the phonemes are known a priori. From this probability of occurrence, entropy of each phrase can be determined using

\[ H = - \sum \pi_i \log_2 \pi_i \]  

(7.1)

Where \( \pi_i \)'s are the probability of occurrence of each phoneme. Eqn.7.1 is a standard expression of entropy of discrete symbol source. \( \pi_i \) stands for probability of occurrence of the symbol. Here however, the significance of \( \pi_i \) and the evaluation of \( \pi_i \) are different. A complete phoneme is treated as a symbol from a source phrase. Thus the contribution towards the entropy of the source (phrase set) due to the phoneme is averaged and determined. For reasons already explained in the preceding chapter, this is speaker dependent.

Similarly entropy is determined for all the four phrases and a phoneme entropy vector is obtained. This phoneme entropy vector is standard for all the speakers. From the actual phoneme timings and transition timings, a Feature Entropy Vector is formed. The ratio of individual phoneme duration to the total phrase duration is calculated for the selected phonemes. Since the phonemes are distributed almost equally throughout the phrases, an average of the ratios of the similar phonemes is determined and substituted for those phonemes. This phoneme
matrix is similar to the probability matrix obtained from the text. Similarly the transition matrices corresponding to the phonemes at the rising and falling regions are also determined. A linear combination of these three matrices yield a single feature matrix. Entropy is now determined for each row in the matrix and feature entropy vector is obtained. The difference between a speaker's feature entropy vector and the common phoneme entropy vector generates a difference vector, which is stored as the reference template for that speaker. Similar templates are created for all the speakers. The test template is also created in the similar fashion and compared with the reference templates stored in the system. The reference template showing minimum deviation from the test template is identified.

7.3 SIMILARITY MEASURE METHOD

In this approach measurements are taken from the actual speech and the similarity between two patterns are determined. The phrases are segmented into phonemes and further to smaller segment of 600 samples. From this phoneme segments, feature measurements are extracted. The time measurements viz., individual phoneme times and two transition times for each phoneme, are determined during the automatic segmentation of the phrases. The other measurements are also taken from each individual phoneme segments. The final set of measurements used for identification is fixed by trial and error method based on
the performance. Different combinations of the extracted features are tried and performance is evaluated. Each set of measurements give a matrix corresponding to the four phrases. Similar matrices are taken for all the speakers and the comparison is made based on a similarity measure. Each element in the matrix is compared with another matrix and weightage is given for every measurement if that falls within a particular limit. The limits for the measurements are also fixed by trial and error method. The weightages are based on the consistency of the measurements. The measurement showing maximum consistency gets more weightage and measurement showing minimum consistency gets least weightage. The weightages are so given that if all the measurements agree for an individual phoneme, the similarity factor is 1. In an ideal case the similarity factor for two similar phrase set is 16. The measurements contributing least to the performance are removed and other measurements are tried. Thus an effective measurement set is fixed after several trials. This gives a good performance.

7.4 CONCLUSION

Two approaches for speaker identification are explained in this chapter. These two approaches are rather simple and easy to implement.